SPECIFICATION

Please amend the specification as indicated herein.

Please amend paragraph [0026] as follows:

[0026] FIG. 15A is a block diagram illustrating an example electronic communication device according to one embodiment of the present invention.

After paragraph [0026] and before paragraph [0027], please add the following new paragraph:

FIG. 15B is a block diagram illustrating an example design of the processor to limit power consumption according to one embodiment of the present invention.

Please amend paragraph [0033] as follows:

[0033] FIG. 4 is a block diagram illustrating the threading module 400 according to one embodiment of the present invention. The threading module 400 to determines which current and previous messages to correlate into threads for display to the user. The threading module 400 of FIG. 4 comprises an SMS message database 410, a threading engine 430, a threading rules database, and an SMS application 440. An operating system such as Palm OS 5 by Palm, Inc., Windows CE or Windows XP by Microsoft, or any other message passing operating system facilitates interoperability between hardware and software, and inter-process communication in the threading module 400. In one embodiment the threading module 400 is implemented in software executed in hardware as shown in FIG. 15A. Other embodiments implement the threading module in hardware or firmware. Additionally, the threading module 400 is coupled to receive SMS messages from an SMS network as shown in FIG. 16.

Please amend paragraph [0064] as follows:

[0064] FIG. 15A is a block diagram illustrating an example electronic communication 1500 device according to one embodiment of the present invention. The electronic communication device 1500, such as a GSM or CDMA cell phone, comprises a transceiver 1510, a SIM (Subscriber Identity Module) card 1520, a processor 1530, an input/output controller 1550, and a memory 1540 each coupled in communication with a conventional bus 1599.

Please amend paragraph [0067] as follows:

[0067] The processor 1530 executes instructions and manipulates data necessary for SMS message threading. The processor 1530 may be a microprocessor such as an ARM processor by ARM, Inc., a Pentium 4® by Intel Corp. or an Athlon XP® by Advanced Micro Devices, an ASIC, a FPD (Field Programmable Device), a microcontroller, or any other device capable of executing instructions and manipulating data. In an embodiment, the processor 1530, comprises a host processor 1532 and a radio processor 1534. The host processor 1532 then executes instructions and manipulates data while the radio processor 1534 communicates with transceiver 1510. Thus, the host processor 1532 can enter into a powered-down, or "sleep," mode to conserve power while the radio processor 1534 continues to operate, allowing transmission and receipt of signals carrying SMS messages in SMS data packets 600.

After paragraph [0067] and before paragraph [0068], please insert the following new paragraphs:

Referring now to Fig. 15B, an exemplary design of the processor 1530 to limiting power consumption will be described. Fig. 15B is a schematic diagram illustrating communication lines between a plurality of processors, the host processor 1532 and the radio processor 1534, according to an exemplary embodiment. As can be seen, a plurality of digital serial ports and control signals 1580 of radio processor 1534 are coupled to host processor 1532. Analog audio signals 1582 are coupled to audio CODEC 1584 for routing to microphones, speakers, or host processor 1532. In one exemplary embodiment, UART1 1585 may be used for multiplexed control and data. UART2 1586 may be used for debug information during development. In an alternative embodiment, UART1 1585 may be used for command and diagnostics information and UART2 1586 may be used for data calls.

In the system of Fig. 5, a plurality of signals are used to control power management functions between host processor 1532 and radio processor 1534.

RESET/ON-OFF (ON/OFF, RADIO_RESET~, RESET_OUT~)

ON/OFF – This active high input signal turns radio processor 1534 on and off. A high level on this pin will turn on radio processor 1534 and boot host processor 1532. A low level on this pin will force radio processor 1534 off immediately, rather than into a graceful shutdown. The graceful shutdown will be accomplished via a software command before the ON/OFF pin is driven low.

RADIO_RESET~ – The active low input pin is the master reset for radio processor 1534. This pin will need to be driven low and then high in order to accomplish a full radio reset.

RESET_OUT~ – This signal is an output from a CDMA radio processor 1534, and indicates that CDMA radio processor 1534 is in RESET and requires initialization.

HANDSHAKE (HOST_WAKE, RADIO_WAKE, HOST_STATUS)

HOST_WAKE – Radio processor 1534 drives this active high signal to host processor 1532 when it wants to wake up host processor 1532 to send a message. In addition, when host processor 1532 wakes up radio processor 1534, this signal is used to acknowledge the RADIO_WAKE signal from host processor 1532. This signal should remain high as long as messages are pending from radio processor 1534 or as long as host processor 1532 has requested radio processor 1534 to be turned on.

RADIO_WAKE – Host processor 1532 drives this active high signal to radio processor 1534 when it wants to wake up radio processor 1534 to send a message. In addition, when radio processor 1534 wakes up host processor 1532, this signal is used to acknowledge the HOST_WAKE signal from radio processor 1534. This signal should remain high as long as messages are pending from host processor 1532 or as long as radio processor 1534 has requested host processor 1532 to be turned on.

HOST_STATUS – This signal is used to inform radio processor 1534 of the sleep status of host processor 1532. When this signal is low, host processor 1532 is asleep, and radio processor 1534 should not send low-priority messages. When the signal is high, host processor 1532 is awake, and radio processor 1534 is free to send message of any priority to host processor 1532.

POWER SUPPLY (VCC_RADIO, VCC_BB, BVATT, GROUND)

VCC_RADIO – This signal is provided by radio processor 1534 and is connected to the I/O voltage of the baseband IC. It will be used by the Duo board to determine when the baseband chip has been powered up and to supply voltage to any interface circuitry, if required.

VCC_BB – This is a power input to radio processor 1534 to be used to derive the various baseband voltages. Rather than use a higher voltage VBATT, this input is a lower voltage (3.2 V typically) to allow for better power efficiency.

VBATT – This is main battery voltage of the system, ranging from approximately 3.4 to 4.2 volts.

GROUND – This is the main ground connection in the system. 4 pins are used for power return, and 2 are used for signal ground.

Handshaking functions can be provided in any of a variety of methods. Exemplary methods are set forth below:

Power On – GSM

1. User powers on device 10.

2. VDDS_OUT is asserted. Host processor 1532 cannot assert MODULE_WAKE until VDSS_OUT is high, in this exemplary embodiment.

3. Radio processor 1534 will come out of reset about 5 ms after VDDS_OUT is high. Radio processor 1534 will then drive HOST_WAKE after initialization is complete and it sees MODULE_WAKE.

4. Radio processor 1534 sends message to host processor 1532 (1st message sent by the modem) to indicate that radio processor 1534 is powered up.

Power On – CDMA

1. User powers on the module.

2. VDDS_OUT is asserted. Host processor 1532 cannot assert MODULE_WAKE until VDDS_OUT is high, in this exemplary embodiment.

3. Radio processor 1534 will come out of reset about 5 ms after VDDS_OUT is high. Radio processor 1534 will then drive HOST_WAKE after initialization is complete and it sees MODULE_WAKE.

4. Host processor 1532 sends Enable_Autonomous_Messaging message to radio processor 1534 (1st message sent by host processor 1532). This message enables radio processor 1534 to send autonomous messages to host processor 1532, and also acts as a soft handshake to signal to radio processor 1534 that host acknowledges that modem is powered up.

5. Radio processor 1534 de-asserts ~RESET_OUT after receiving the message.

Power Off

1. User powers down the module.

2. CPU sends graceful deregistration commands to radio processor 1534 and waits for confirmation response.

3. CPU de-asserts ON/OFF which immediately shuts off power to radio processor 1534.

Host Sleep/Wake Status

The CDMA radio processor 1534 needs to know host processor's 1532 sleep mode status to suppress low priority messages (GSM radio module does not). This is important to conserve host's power by shutting down the UARTs. When host goes to sleep, it will de-assert the HOST_STATUS signal, which will result in an interrupt to radio processor 1534. Modem will suppress all low priority control messages until this signal is asserted.

Host sends data to Modem

1. Host asserts MODULE_WAKE interrupt to signal to radio processor 1534 that it wishes to initiate data transfer. Host UART is already enabled at this point.

2. Modem woke up if it was in sleep mode. TCXO, UART1 and UART2 are enabled. Modem is ready to receive data. It asserts HOST_WAKE interrupt to signal to host processor 1532 that it's ready to receive data.

3. Message transfer occurs. Multiple messages can be exchanged.

4. After the message has terminated, there is a short hysteresis period to prevent unnecessary toggling of either of the handshake signals.

5. Host de-asserts MODULE_WAKE interrupt since it has finished all the message transfer at this point. Modem however can still send messages to host processor 1532. Modem will NOT sleep until MODULE_WAKE is de-asserted so host processor 1532 de-asserts MODULE_WAKE when it is finished with its transmission.

6. Modem de-asserts HOST_WAKE interrupt if it also does not have any other messages to send. Note: The initiator (host in this case) does not have to terminate the communication first. Now since both MODULE_WAKE and HOST_WAKE are de-asserted, the communication channel is closed and UART clocks are disabled.

7. Both processors can enter into sleep state.

Modem sends data to Host

1. Modem asserts HOST_WAKE interrupt to signal to host processor 1532 that it wishes to initiate data transfer. Modem UARTs are already enabled at this point.

2. Host wakes up if it was in sleep mode. Host's UARTs are enabled. Host is ready to receive data on the UARTs. Host asserts MODULE_WAKE to signal to radio processor 1534 that it's ready to receive data.

3. Message transfer occurs. Multiple bi-directional messages can be exchanged.

4. After the message is terminated, there is a short hysteresis period to prevent unnecessary toggling of either of the handshake signals.

5. Modem de-asserts HOST_WAKE interrupt since it has finished all the message transfer at this point. Host can still send messages.

6. Host de-asserts MODULE_WAKE interrupt since it's done with message transfer. The communication channel is closed.

7. Both processors can enter into sleep state.

Thus, in one embodiment, the processor 1530 can be configured to conserve

power use by allowing the host processor 1532 to enter a reduced power mode when data signals are not being processed, upon receiving a data singal to be processed, the radio processor 1534 can then wake the host processor 1532. Alternatively, the radio processor 1534 can enter a reduced power mode when there are no data signals being transmitted or received, the host processor 1532 can then wake the radio processor 1534 when a data signal is ready for transmission.